ADDENDUM

MITS Hard Disk Preliminary Documentation

The following addendum contains information about the Hard Disk controller and associated firmware. It includes the following topics:

1. Hard Disk Sector format - the arrangement of information in a disk sector.
2. Controller Commands - the format and action of the controller's firmware commands.
3. Error codes - the meaning of the error flags that are transmitted to the computer through I/O port 161 octal.
4. Firmware Data Flow Diagram - shows the flow of data and status signals between the computer and the controller.
5. Firmware Listing - source listing of the controller program for the 8X300 interpreter.
6. Understanding the 8X300 Instruction Set.

1. HARD DISK SECTOR FORMAT

Data stored on the hard disk is divided into sectors. The format of a sector is as follows:

Contents	Length
Header (see below)	
Sync byte (255_{10})	1 byte
Data	256 bytes
CRC	2 bytes

The header has the following format:

Contents	Length
Preamble	*
Sync byte (255_{10})	1 byte
Cylinder address	2 bytes
Head # + Sector #	1 byte**
Header CRC	2 bytes
Header gap	*

* The preamble and the header gap are strings of approximately 26 bytes of zeros. They provide a buffer zone to separate the sectors and allow for small head positioning errors.

** The head number is the high order three bits of the byte; the remainder is the sector number. The head numbers and their associated bit patterns are as follows:

Head Number	Bits 5 - 7
0	100
1	101
2	110
3	111

Head Number	Bits 5 - 7
4	000
5	001
6	010
7	011

2. COMMANDS

Command	Port	Bit 7	6	5	4	3	2	1	0
SEEK	163	0	0	0	0	drive		X	High bit of cylinder address
	167	low byte of cylinder address							
WRITE SECTOR	163	0	0	1	0	drive		buffer	
	167	head				sector			
READ SECTOR (FORMATTED)	163	0	0	1	1	drive		buffer	
	167	head				sector			
READ BUFFER	163	0	1	0	1	X	X	buffer	
	167	byte count							
WRITE BUFFER	163	0	1	0	0	X	X	buffer	
	167	byte count							

If the byte read from port 167 in the WRITE BUFFER and READ BUFFER commands is zero, the byte count is assumed to be 256.

READ STATUS	163	0	1	1	0	drive		X	X
	167	status word address							
SET IV BYTE	163	1	0	0	0	X	X	X	X
	167	IV byte address							

This command reads port 167 twice. The first byte read is the address of the IV byte to be set. The second byte read is the data to be deposited in the selected IV byte.

READ UNFORMATTED	163	1 0 1 0 | drive | buffer
	167	head | sector
FORMAT	163	1 1 0 0 | drive | X X
	167	head | X X X X X
INITIALIZE	163	1 1 1 0 | X X X X

3. ERROR FLAGS

Port 161

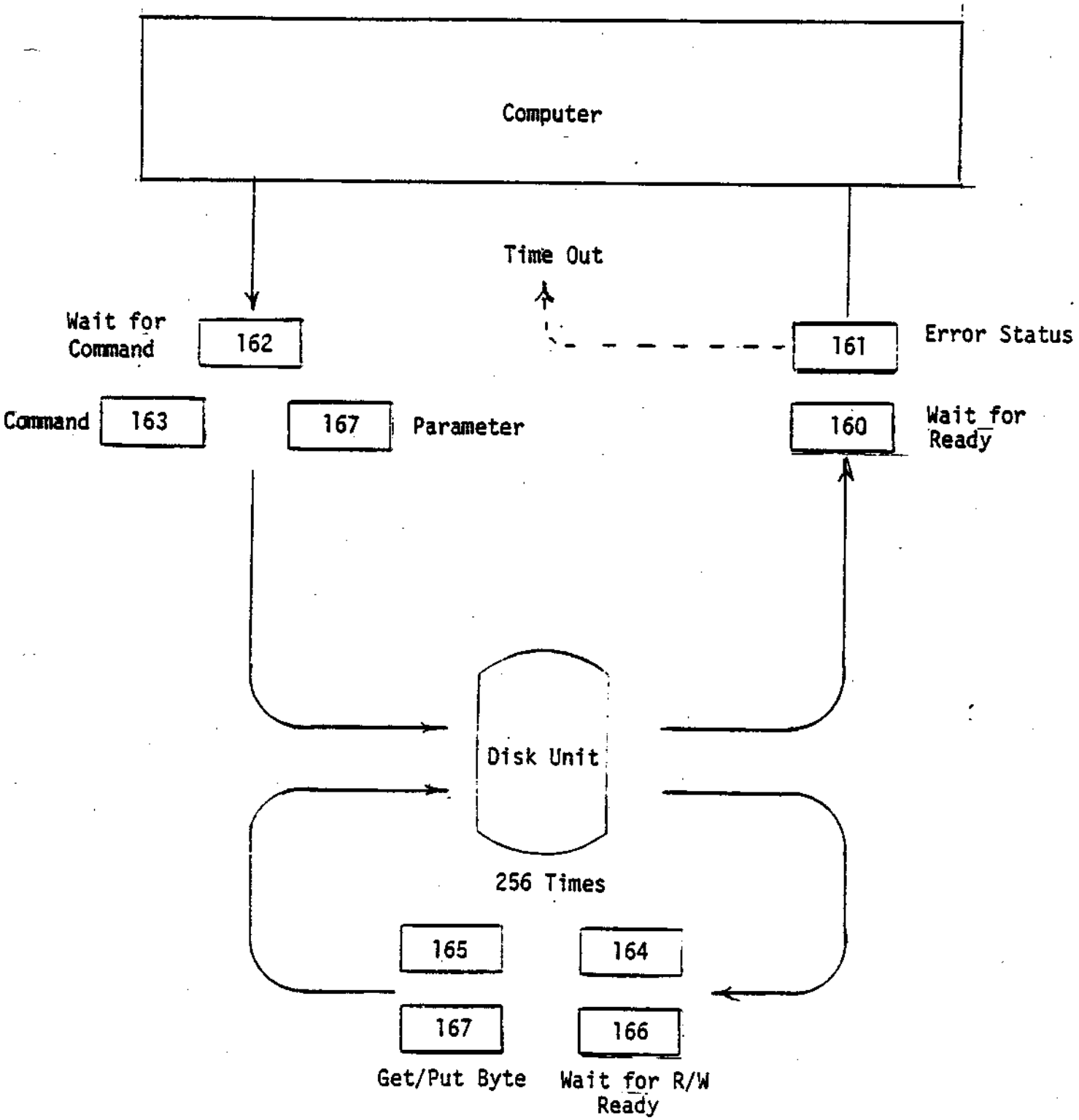
Bit	1 means:[1]	Error may occur in:
0	Drive not ready	Any command except INITIALIZE and SET IV BYTE
1	Illegal Sector	SEEK, READ SECTOR, WRITE SECTOR, FORMAT and READ UNFORMATTED commands
2	CRC error in sector read	READ SECTOR, READ UNFORMATTED[2] commands
3	CRC error in header read	SEEK, READ SECTOR, WRITE SECTOR and FORMAT commands
4	Header has wrong sector	Same as in bit 3
5	Header has wrong cylinder	Same as in bit 3[3]
6	Header has wrong head	Same as in bit 3
7	Write Protect	Same as in bit 1[4]

NOTES

1. All bits of port 161 are ones on the first read after the controller is turned on.
2. Always occurs on an unformatted read of a formatted sector.
3. Occurs spuriously when one of these commands is issued for a drive different from the one specified in the last seek. This spurious error is ignored by the write logic.
4. Only relevant during a Write Sector command. If a sector is write-protected, data may not be written into it. The Write Sector command is ignored and the error flag is set.

Error	Explanation
Drive not ready	Something has caused the drive to go off line. Check the drive number switch and the ready light.
CRC error in sector read	A Cyclical Redundancy Check exception was detected while the 256 data bytes of a sector were being read.
CRC error in header read.	A Cyclical Redundancy Check exception was detected while the header of a sector was being read. If the error persists, the sector is probably unusable. Reformat the disk
Header has wrong sector	The header did not contain the expected sector number. Indicates an anomaly in disk rotation. Try again.
Header has wrong cylinder	The header did not contain the expected cylinder address. Indicates a head positioning error. Try again.
Header has wrong head #	The header did not contain the expected head number. Indicates a problem in the drive electronics.
Write protect.	A command was issued for a write-protected sector. If the command was a Write Sector command, it is ignored. Other commands are executed normally.

4. FIRMWARE DATA FLOW DIAGRAM



The arrows show the direction of information flow. The boxes represent computer I/O ports with the octal addresses noted inside. The labels outside the I/O port boxes indicate the type of information transmitted through the ports.

For example, the controller waits for the presence of a command to be indicated through port 162. It then retrieves the command code from port 163 and the parameter(s) of the command from port 167.

```
OBJ      OBJECT WORD      SOURCE        SOURCE LINE                COMMENT
LOC                       NBR

0000  ........ ........  0001  *
0000  ........ ........  0002  **      VERSION 4.0 OF TCS CONTROLLER FIRMWARE
0000  ........ ........  0003  **      THIS VERSION INCLUDES HEADER SUPPORT
0000  ........ ........  0004  **      MODIFIED FROM V3.27 BY H. ROUSSO
0000  ........ ........  0005  **
0000  ........ ........  0006  *
0000  ........ ........  0007  **      INITIALIZATION (ENTERED BY CONTROLLER RESET)
0000  ........ ........  0008  *
0000  ........ ........  0009        ORG     0
0000  ........ ........  0010  START EQU     *
0000  11000001 11111111  0011        XMIT    R1,255          INIT DRIVE INTERFACE TO FALSE
0001  11000111 00010011  0012        XMIT    IVL,19          TRACK 0
0002  00000001 00010111  0013        MOVE    R1,IV
0003  11000111 00010010  0014        XMIT    IVL,18
0004  00000001 00010111  0015        MOVE    R1,IV
0005  11000111 00010001  0016        XMIT    IVL,17
0006  00000001 00010111  0017        MOVE    R1,IV
0007  11010111 00100000  0018        XMIT    IV(7,1),0       SELECT DRIVE ZERO
0010  11000111 00010010  0019        XMIT    IVL,18
0011  11010100 00100000  0020        XMIT    IV(4,1),0       SLOW SEEK TO TRACK 0
0012  11010101 00100000  0021        XMIT    IV(5,1),0
0013  11010101 01000011  0022        XMIT    IV(5,2),3
0014  11000111 00000001  0023        XMIT    IVL,1
0015  11000000 00010101  0024        XMIT    AUX,21          SET 2,4,6 OUT; 3,5,7 IN
0016  00000000 00010111  0025        MOVE    AUX,IV
0017  11000111 00000100  0026        XMIT    IVL,4           RESET HANDSHAKE LINES
0020  00000001 00010111  0027        MOVE    R1,IV
0021  11000111 00010101  0028        XMIT    IVL,21          WAIT FOR SEEKS TO FINISH
0022  10110111 10010010  0029        NZT     IV(7,4),*
0023  ........ ........  0030  *     END OF INITIALIZATION
0023  ........ ........  0031  */PAGE/
0023  ........ ........  0032  *
0023  ........ ........  0033  **      GET COMMAND FROM ALTAIR
0023  ........ ........  0034  *
0023  ........ ........  0035  COMM    EQU     *
```

```
OBJ LOC   OBJECT WORD        SOURCE NBR   SOURCE LINE                    COMMENT

0023  11000111 00000100  0036          XMIT   IVL,4              STROBE CONTROLLER READY
0024  11010111 00100000  0037          XMIT   IV(7,1),0
0025  11010111 00100001  0038          XMIT   IV(7,1),1
0026  11000111 00000101  0039          XMIT   IVL,5
0027  10110110 00110111  0040          NZT    IV(6,1),*          WAIT FOR COMMAND
0030  11000111 00000011  0041          XMIT   IVL,3              GET HIGH COMMAND BYTE
0031  00010010 01100000  0042          MOVE   IV(2,3),AUX        GET COMMAND CLASS
0032  00010111 10100001  0043          MOVE   IV(7,5),R1         GET COMMAND SUBTYPE
0033  11000111 00000111  0044          XMIT   IVL,7
0034  00010111 00000010  0045          MOVE   IV,R2              AND ANY PARMS
0035  11000111 00000100  0046          XMIT   IVL,4
0036  11010110 00100000  0047          XMIT   IV(6,1),0          SEND COMMAND ACK
0037  11010110 00100001  0048          XMIT   IV(6,1),1
0040  11000111 00000010  0049          XMIT   IVL,2              CLEAR ERROR WORD
0041  11000110 00000000  0050          XMIT   R6,0
0042  00000110 00010111  0051          MOVE   R6,IV
0043  11001001 00000001  0052          XMIT   R11,1              SET UP RETURN VECTOR
0044  10000000 00100101  0053          XEC    AUX,COMTAB
0045  ........ ........  0054  COMTAB  EQU    *
0045  11100000 00101101  0055          JMP    SEEK               SEEK CYL (FORMATTED)
0046  11100000 01000010  0056          JMP    SECTOR             TRANSFER SECTOR (FORMATTED)
0047  11100000 01001101  0057          JMP    BUFFER             TRANSFER BUFFER
0050  11100000 01110010  0058          JMP    STATUS             READ STATUS
0051  11100000 01111110  0059          JMP    SETBYTE            SET IV BYTE
0052  11100000 10001000  0060          JMP    READUN             READ SECTOR (UNFORMATTED)
0053  11100000 10010001  0061          JMP    FORMAT             FORMAT DISK
0054  11100000 00000000  0062          JMP    START              UNIMPLEMENTED
0055  ........ ........  0063  */PAGE/
0055  ........ ........  0064  *
0055  ........ ........  0065  **      EXECUTE SEEK COMMAND
0055  ........ ........  0066  *
0055  11000000 11111111  0067  SEEK    XMIT   AUX,255             MOVE COMPLEMENT
0056  11000111 00010011  0068          XMIT   IVL,19
0057  01100010 00010111  0069          XOR    R2,IV              SEND LOW ORDER CYL ADDR
0060  11000111 00010010  0070          XMIT   IVL,18
```

```
OBJ LOC   OBJECT WORD        SOURCE NBR  SOURCE LINE                          COMMENT
0061   01100001 00110111     0071        XOR     R1,IV(7,1)                   AND CYL MSB
0062   11000010 00000000     0072        XMIT    R2,0                         SELECT HEAD# 0
0063   11100001 10110000     0073        JMP     SELECT
0064   11000000 00000001     0074  SEEK1 XMIT    AUX,1                        FOR INCREMENT
0065   11000111 00010110     0075        XMIT    IVL,22                       GET SECTOR COUNTER
0066   00110111 10100010     0076        ADD     IV(7,5),R2                   ... PLUS 1
0067   11000000 00011000     0077        XMIT    AUX,24                       CHECK FOR WRAPAROUND
0070   01100010 00000000     0078        XOR     R2,AUX
0071   10100000 00111011     0079        NZT     AUX,SEEK2                    NO WRAP AROUND
0072   11000010 00000000     0080        XMIT    R2,0                         WRAP TO SECTOR 0
0073   11000111 00010010     0081  SEEK2 XMIT    IVL,18
0074   11010101 00100000     0082        XMIT    IV(5,1),0                    STROBE...
0075   11100000 00111110     0083        NOP                                  ...(DELAY)
0076   11010101 00100001     0084        XMIT    IV(5,1),1                    ...SEEK LINE
0077   11000111 00010101     0085        XMIT    IVL,21                       WAIT FOR SEEKS TO SETTLE
0100   10110111 10000000     0086        NZT     IV(7,4),*
0101   11100000 11001010     0087        JMP     READHED
0102   ........ ........     0088  *
0102   ........ ........     0089  */PAGE/
0102   ........ ........     0090  *
0102   ........ ........     0091  **      EXECUTE READ/WRITE DISK SECTOR
0102   ........ ........     0092  **      BEFORE READING OR WRITING, CHECK HEADER:
0102   ........ ........     0093  **      COMPARES CYL ADDR (INTERFACE IV BYTES 18,19);
0102   ........ ........     0094  **              SECTOR ( R2);
0102   ........ ........     0095  **              HEADER CRC.
0102   ........ ........     0096  **      ERROR STATUS GOES INTO IV BYTE 2, BITS 2,3,4.
0102   ........ ........     0097  **      IF HEADER ERROR, WILL NOT DO WRITE
0102   ........ ........     0098  **       BUT WILL READ SECTOR ANYWAY
0102   ........ ........     0099  *
0102   11001001 00000010     0100  SECTOR  XMIT    R11,2                        RETURN VECTOR
0103   11000111 00000001     0101  SECT1   XMIT    IVL,1                        SELECT BUFFER (READUN JMPS HERE)
0104   00000001 01010001     0102        MOVE    R1,IV(1,2)
0105   11100001 10110000     0103        JMP     SELECT
0106   ........ ........     0104  SECT2   EQU     *
0107   00000001 00000110     0105        MOVE    R1,R6                        STORE R/W BIT
```

```
OBJ       OBJECT WORD      SOURCE          SOURCE LINE                COMMENT
LOC                        NBR

0107   11100000 11001010   0106            JMP     READHED            CHECK HEADER
0110   ........ ........   0107   SECT3    EQU     *
0110   11000000 00000001   0108            XMIT    AUX,1              R/W MASK BIT
0111   01000110 01000001   0109            AND     R6(2),R1           SHIFT AND MASK
0112   10000001 01001011   0110            XEC     R1,*+1             0 = WRITE, 1 = READ
0113   11100001 10000010   0111            JMP     WRITSUB
0114   11100001 00011111   0112            JMP     READSUB
0115   ........ ........   0113   */PAGE/
0115   ........ ........   0114   *
0115   ........ ........   0115   **      EXECUTE TRANSFER BUFFER COMMAND
0115   ........ ........   0116   **      THIS ROUTINE TRANSFERS A BUFFER FROM OR TO THE ALTAIR
0115   ........ ........   0117   *
0115   ........ ........   0118   BUFFER   EQU     *
0115   11000111 00000001   0119            XMIT    IVL,1              NOTE: NOT CALLABLE
0116   00000001 01010001   0120            MOVE    R1,IV(1,2)         SELECT BUFFER
0117   11000000 11111111   0121            XMIT    AUX,255            GET 2'S COMPLEMENT OF BYTE COUNT
0120   01100010 00000010   0122            XOR     R2,R2
0121   11000000 00000001   0123            XMIT    AUX,1
0122   00100010 00000010   0124            ADD     R2,R2
0123   01000001 10000001   0125            AND     R1(4),R1           GET DIRECTION
0124   11000011 00000000   0126            XMIT    R3,0               INITIALIZE WS PTR
0125   10100001 01100100   0127            NZT     R1,RDBUFF          IF NONZERO, MUST BE READ
0126   ........ ........   0128   WRBUFF   EQU     *
0126   11000111 00000100   0129            XMIT    IVL,4
0127   11010000 00100000   0130            XMIT    IV(0,1),0          STROBE...
0130   11010000 00100001   0131            XMIT    IV(0,1),1          ...DATA READY LINE
0131   ........ ........   0132   WRB1     EQU     *
0131   00000011 00001111   0133            MOVE    R3,IVR             SELECT WS LOC
0132   11000111 00000101   0134            XMIT    IVL,5
0133   10110000 00111011   0135            NZT     IV(0,1),*          WAIT ON ALTAIR DATA STROBE
0134   11000111 00000111   0136            XMIT    IVL,7
0135   00010111 00011111   0137            MOVE    IV,WS
0136   00100011 00000011   0138            ADD     R3,R3              MOVE WS PTR
0137   00100010 00000010   0139            ADD     R2,R2              INCREMENT BYTE COUNT
0140   11000111 00000101   0140            XMIT    IVL,5              MAKE SURE DATA STROBE RESET
```

```
OBJ       OBJECT WORD       SOURCE          SOURCE LINE                COMMENT
LOC                         NBR

0141  10010000 00100001  0141            XEC     IV(0,1),*
0142  10100010 01011001  0142            NZT     R2,WRB1
0143  11100000 00010011  0143            JMP     COMM
0144  ........ ........  0144  *
0144  ........ ........  0145  **        READ BUFFER
0144  ........ ........  0146  *
0144  ........ ........  0147  RDBUFF    EQU     *
0144  11000111 00000100  0148            XMIT    IVL,4              STROBE...
0145  11010001 00100000  0149            XMIT    IV(1,1),0          ...
0146  11010001 00100001  0150            XMIT    IV(1,1),1          ...DATA READY LINE
0147  ........ ........  0151  RDB1      EQU     *
0147  00000011 00001111  0152            MOVE    R3,IVR             SET WS PTR
0150  11000111 00000110  0153            XMIT    IVL,6
0151  00011111 00010111  0154            MOVE    WS,IV              WRITE BYTE TO ALTAIR
0152  00100011 00000011  0155            ADD     R3,R3              MOVE WS PTR
0153  00100010 00000010  0156            ADD     R2,R2              MOVE BYTE CTR
0154  11000111 00000101  0157            XMIT    IVL,5
0155  10010001 00101101  0158            XEC     IV(1,1),*
0156  10110001 00101110  0159            NZT     IV(1,1),*          WAIT ON STROBE TO GO LOW
0157  10010001 00101111  0160            XEC     IV(1,1),*          THEN HIGH AGAIN
0160  10100010 01100111  0161            NZT     R2,RDB1            NOT DONE = BRANCH
0161  11100000 00010011  0162            JMP     COMM
0162  ........ ........  0163  */PAGE/
0162  ........ ........  0164  *
0162  ........ ........  0165  **        EXECUTE THE STATUS TRANSFER COMMAND
0162  ........ ........  0166  *
0162  ........ ........  0167  STATUS    EQU     *
0162  11001001 00000101  0168            XMIT    R11,5              GET RETURN VECTOR
0163  00000010 00000011  0169            MOVE    R2,R3              SAVE IV-BYTE ADDRESS
0164  11000010 00000000  0170            XMIT    R2,0               SET UP HEAD# 0 FOR SELECT
0165  11100001 10110000  0171            JMP     SELECT             SELECT HEAD,DRIVE
0166  ........ ........  0172  STAT1     EQU     *
0166  00000011 00000111  0173            MOVE    R3,IVL             SELECT STATUS WORD
0167  00010111 00000010  0174            MOVE    IV,R2              READ IT
0170  11000111 00000110  0175            XMIT    IVL,6
```

```
OBJ LOC   OBJECT WORD        SOURCE NBR  SOURCE LINE                          COMMENT
0171  00000010 00010111  0176          MOVE    R2,IV              SEND IT TO ALTAIR
0172  11000111 00000100  0177          XMIT    IVL,4
0173  11010001 00100000  0178          XMIT    IV(1,1),0          STROBE...
0174  11010001 00100001  0179          XMIT    IV(1,1),1          ...DONE
0175  11100000 00010011  0180          JMP     COMM
0176  ........ ........  0181  */PAGE/
0176  ........ ........  0182  *
0176  ........ ........  0183  **      EXECUTE SET IV BYTE DATA COMMAND
0176  ........ ........  0184  *
0176  ........ ........  0185  SETBYTE EQU     *
0176  11000111 00000100  0186          XMIT    IVL,4              SEND RDY FOR DATA FLAG
0177  11010000 00100000  0187          XMIT    IV(0,1),0
0200  11010000 00100001  0188          XMIT    IV(0,1),1
0201  11000111 00000101  0189          XMIT    IVL,5              WAIT FOR ALTAIR DATA STROBE
0202  10110000 00100010  0190          NZT     IV(0,1),*
0203  11000111 00000111  0191          XMIT    IVL,7              GET DATA VALUE
0204  00010111 00000011  0192          MOVE    IV,R3
0205  00000010 00000111  0193          MOVE    R2,IVL             SELECT DESIRED IV BYTE
0206  00000011 00010111  0194          MOVE    R3,IV
0207  11100000 00010011  0195          JMP     COMM               END OF COMMAND
0210  ........ ........  0196  */PAGE/
0210  ........ ........  0197  *
0210  ........ ........  0198  **      EXECUTE UNFORMATTED SECTOR READ
0210  ........ ........  0199  *
0210  ........ ........  0200  READUN  EQU     *
0210  11001001 00000100  0201          XMIT    R11,4              RETURN PARM
0211  11100000 01000011  0202          JMP     SECT1              DO COMMON STUFF & RETURN HERE
0212  ........ ........  0203  READU1  EQU     *
0212  11000100 00011000  0204          XMIT    R4,24              LOW ORDER FOR 257+CRC
0213  11000011 00001000  0205          XMIT    R3,8               HIGH ORDER FOR 257+CRC
0214  11000000 00000001  0206          XMIT    AUX,1              READ MODE
0215  11100001 11001100  0207          JMP     DDSET
0216  11100001 11101010  0208  READU2  JMP     SECTW
0217  ........ ........  0209  READU3  EQU     *
0217  11000001 01010000  0210          XMIT    R1,80              COUNT FOR 1/2 PREAMBLE
```

```
OBJ LOC   OBJECT WORD   SOURCE NBR  SOURCE LINE                    COMMENT

0220  11100001 00100101  0211          JMP     RDS11              JUMP TO READ SUBROUTINE
0221  ........ ........  0212  */PAGE/
0221  ........ ........  0213  *
0221  ........ ........  0214  **      EXECUTE THE FORMAT DISK COMMAND
0221  ........ ........  0215  **
0221  ........ ........  0216  **      FORMATS THE SELECTED HEAD # WITH HEADER INFORMATION:
0221  ........ ........  0217  **      PREAMBLE, SYNC BYTE (ALL 1'S),
0221  ........ ........  0218  **      CYL ADDR (2 BYTES, MSB FIRST),
0221  ........ ........  0219  **      HEAD+SECTOR # (1 BYTE)
0221  ........ ........  0220  **      HEADER CRC (2 BYTES),
0221  ........ ........  0221  **      HEADER GAP
0221  ........ ........  0222  **
0221  ........ ........  0223  **      AFTER WRITING, GOES BACK AND CHECKS HEADER.
0221  ........ ........  0224  **      ERROR STATUS LOADED INTO IV BYTE 2
0221  ........ ........  0225  *
0221  ........ ........  0226  FORMAT  EQU     *
0221  11001001 00000011  0227          XMIT    R11,3              RETURN PARM FOR SELECT
0222  11100001 10110000  0228          JMP     SELECT             SELECT DRIVE
0223  ........ ........  0229  FORMS   EQU     *
0223  11001001 00000010  0230          XMIT    R11,2              RETURN VECTOR FOR WRITHED
0224  ........ ........  0231  FORM0   EQU     *
0224  11000000 11111111  0232          XMIT    AUX,255            INIT CYL ADDR TO ZERO
0225  11000111 00010011  0233          XMIT    IVL,19
0226  00000000 00010111  0234          MOVE    AUX,IV
0227  11000111 00010010  0235          XMIT    IVL,18
0230  00000000 00010111  0236          MOVE    AUX,IV
0231  11010100 00100000  0237          XMIT    IV(4,1),0          SET RESTORE
0232  11010101 00100000  0238          XMIT    IV(5,1),0          SET CYL STROBE
0233  11100000 10011100  0239          NOP
0234  11010101 01000011  0240          XMIT    IV(5,2),3          RESET CYL STROBE, RESTORE
0235  11000111 00010101  0241  FORM05  XMIT    IVL,21             WAIT FOR SEEK TO SETTLE
0236  10110111 10011110  0242          NZT     IV(7,4),*
0237  11000110 00000000  0243          XMIT    R6,0               INITIALIZE SECTOR #
0240  ........ ........  0244  FORM1   EQU     *
0241  00000110 00000010  0245          MOVE    R6,R2              READHED WANTS SECTOR # R2
```

```
OBJ LOC   OBJECT WORD          SOURCE NBR   SOURCE LINE                          COMMENT
0241   10001001 10100000   0246           XEC     R11,*-1
0242   11100001 01001000   0247           JMP     WRITHED          WRITE HEADER
0243   11100000 11001010   0248           JMP     READHED          READ HEADER
0244   ........ ........   0249  FORM20   EQU     *
0244   11000111 00000010   0250           XMIT    IVL,2            CHECK FOR HEADER READ ERROR
0245   11000000 00000111   0251           XMIT    AUX,7            ERROR MASK
0246   01010100 01100000   0252           AND     IV(4,3),AUX
0247   10100000 11000111   0253           NZT     AUX,FORM61       IF ERROR, STOP READING
0250   ........ ........   0254  FORM21   EQU     *
0250   11000000 00000001   0255           XMIT    AUX,1            INCREMENT VALUE
0251   00100110 00000110   0256           ADD     R6,R6            INCREMENT SECTOR #
0252   11000000 00011000   0257           XMIT    AUX,24           CHECK IF LAST SECTOR+1
0253   01100110 00000000   0258           XOR     R6,AUX
0254   10100000 10100000   0259           NZT     AUX,FORM1                REPEAT IF MORE
0255   11000111 00010011   0260           XMIT    IVL,19           LOAD CYL LSB'S INTO R2
0256   11000000 11111111   0261           XMIT    AUX,255          COMPLEMENT
0257   01110111 00000010   0262           XOR     IV,R2
0260   11000111 00010010   0263           XMIT    IVL,18           CHECK CYL MSB
0261   10110111 00110110   0264           NZT     IV(7,1),FORM3    IF FALSE, IGNORE CYL LSB
0262   11000000 10010101   0265           XMIT    AUX,149          CYL ADDR 405 (8 LSB)
0263   01100010 00000000   0266           XOR     R2,AUX           CHECK FOR LAST CYL ADDR
0264   10100000 10110110   0267           NZT     AUX,FORM3        IF NOT LAST, INC CYL ADDR
0265   11100000 11000100   0268           JMP     FORM6            CHECK MODE
0266   ........ ........   0269  FORM3    EQU     *
0266   11000000 00000001   0270           XMIT    AUX,1
0267   00100010 00000010   0271           ADD     R2,R2            INC CYL ADDR
0270   10101000 10111010   0272           NZT     OVF,FORM4        CHECK OVERFLOW
0271   11100000 10111100   0273           JMP     FORM5            NO OVERFLOW - LOAD LSB
0272   ........ ........   0274  FORM4    EQU     *
0272   11000111 00010010   0275           XMIT    IVL,18           OVERFLOW,...
0273   11010111 00100000   0276           XMIT    IV(7,1),0        ...SET CYL MSB TRUE
0274   11000111 00010011   0277  FORM5    XMIT    IVL,19
0275   11000000 11111111   0278           XMIT    AUX,255          SEND CYL LSB TO INTERFACE
0276   01100010 00010111   0279           XOR     R2,IV            COMPLEMENT
0277   11000111 00010010   0280           XMIT    IVL,18           PULSE CYL STROBE
```

```
OBJ LOC   OBJECT WORD        SOURCE NBR  SOURCE LINE                          COMMENT
0300  11010101 00100000  0281          XMIT    IV(5,1),0
0301  11100000 11000010  0282          NOP
0302  11010101 00100001  0283          XMIT    IV(5,1),1
0303  11100000 10011101  0284          JMP     FORM05              NEXT TRACK,SECTOR 0
0304  11000000 00000011  0285  FORM6   XMIT    AUX,3               FINISHED ALL CYLS
0305  01101001 00000000  0286          XOR     R11,AUX             READING OR WRITING ?
0306  10100000 11001000  0287          NZT     AUX,FORM7           1 = FINISHED WRITING
0307  ........ ........  0288  FORM61  EQU     *
0307  11100000 00010011  0289          JMP     COMM                FINISHED READING = RETURN
0310  ........ ........  0290  FORM7   EQU     *
0310  11001001 00000011  0291          XMIT    R11,3               RETURN VECTOR FOR READ MOCE
0311  11100000 10010100  0292          JMP     FORM0               GO BACK & READ HEADERS
0312  ........ ........  0293  */PAGE/
0312  ........ ........  0294  *
0312  ........ ........  0295  **      READ HEADER SUBROUTINE
0312  ........ ........  0296  **
0312  ........ ........  0297  **      THIS READS THE HEADER AND COMPARES THE CYL ADDR
0312  ........ ........  0298  **      WITH THAT STORED IN INTERFACE IV BYTES 18 AND 19,
0312  ........ ........  0299  **      COMPARES SECTOR # WITH R2 (5 LSB),
0312  ........ ........  0300  **      AND CHECKS THE HEADER CRC.
0312  ........ ........  0301  **      ERROR STATUS IS STORED IN IV BYTE 2, BITS 2-4
0312  ........ ........  0302  **      BIT 2 - CYL ADDR,
0312  ........ ........  0303  **      BIT 3 - SECTOR #,
0312  ........ ........  0304  **      BIT 4 - HEADER CRC
0312  ........ ........  0305  *
0312  ........ ........  0306  READHED EQU     *
0312  00001001 00000101  0307          MOVE    R11,R5              SAVE RETURN VECTOR
0313  11000100 00110000  0308          XMIT    R4,48               LOW ORDER 4 HEADER+CRC
0314  11000011 00000000  0309          XMIT    R3,0                HIGH ORDER 4+CRC
0315  11001001 00000011  0310          XMIT    R11,3               RETURN VECTOR
0316  11000000 00000001  0311          XMIT    AUX,1               READ MODE
0317  11100001 11001100  0312          JMP     DDSET
0320  11100001 11101010  0313  READH1  JMP     SECTW
0321  11000001 01010000  0314  READH2  XMIT    R1,80               COUNT FOR 1/2 PREAMBLE (40 US)
0 2   11000000 11111111  0315          XM(     AUX,255
```

```
OBJ       OBJECT WORD      SOURCE        SOURCE LINE                COMMENT
LOC                        NBR

0323  00100001 00000001  0316          ADD     R1,R1
0324  10100001 11010011  0317          NZT     R1,*-1
0325  11000111 00100101  0318          XMIT    IVL,37             START TRANSFER
0326  11010001 00100001  0319          XMIT    IV(1,1),1
0327  11000111 00100100  0320          XMIT    IVL,36             WAIT ON DATA READY
0330  10010111 00111000  0321          XEC     IV(7,1),*
0331  11000111 00100101  0322          XMIT    IVL,37             STROBE IT OUT (SYNC BYTE)
0332  11010110 00100001  0323          XMIT    IV(6,1),1
0333  11010110 00100000  0324          XMIT    IV(6,1),0
0334  11000111 00100100  0325          XMIT    IVL,36             WAIT ON DATA READY
0335  10010111 00111101  0326          XEC     IV(7,1),*
0336  11000111 00100001  0327          XMIT    IVL,33             READ HEADER CYL MSB
0337  00010111 00100001  0328          MOVE    IV(7,1),R1
0340  11000000 00000001  0329          XMIT    AUX,1              INVERT CONTROLLER CYL MSB
0341  11000111 00010010  0330          XMIT    IVL,18
0342  01110111 00100000  0331          XOR     IV(7,1),AUX
0343  01100001 00000000  0332          XOR     R1,AUX             COMPARE
0344  11000111 00000010  0333          XMIT    IVL,2              SELECT IV BYTE 2
0345  11100000 11100110  0334          NOP                        ALIGN ADDR. REF. BELOW
0346  00000000 00110010  0335          MOVE    AUX,IV(2,1)        SET BIT 2 IF ERROR
0347  11000111 00100101  0336          XMIT    IVL,37             STROBE IT OUT
0350  11010110 00100001  0337          XMIT    IV(6,1),1
0351  11010110 00100000  0338          XMIT    IV(6,1),0
0352  11000111 00100100  0339          XMIT    IVL,36             WAIT ON DATA READY
0353  10010111 00101011  0340          XEC     IV(7,1),*
0354  11000111 00100001  0341          XMIT    IVL,33             READ HEADER CYL LSB
0355  00010111 00000001  0342          MOVE    IV,R1
0356  11000000 11111111  0343          XMIT    AUX,255            INVERT CONTRL. CYL LSB
0357  11000111 00010011  0344          XMIT    IVL,19
0360  01110111 00000000  0345          XOR     IV,AUX
0361  01100001 00000000  0346          XOR     R1,AUX             COMPARE
0362  10100000 11110100  0347          NZT     AUX,READH3
0363  11100000 11110110  0348          JMP     READH4             EQUAL
0364  ........ ........  0349  READH3  EQU     *
0364  11000111 00000010  0350          XMIT    IVL,2              SET...
```

```
OBJ LOC   OBJECT WORD    SOURCE NBR  SOURCE LINE                          COMMENT
0365  11010010 00100001  0351        XMIT   IV(2,1),1                    ...IV BYTE 2 BIT 2 TO 1
0366  ........ ........  0352 READH4 EQU    *
0366  11000111 00100101  0353        XMIT   IVL,37                       STROBE IT OUT
0367  11010110 00100001  0354        XMIT   IV(6,1),1
0370  11010110 00100000  0355        XMIT   IV(6,1),0
0371  11000111 00100100  0356        XMIT   IVL,36                       WAIT ON DATA READY
0372  10010111 00111010  0357        XEC    IV(7,1),*
0373  11000000 00011111  0358        XMIT   AUX,31                       MASK FOR 5 BIT SECTOR
0374  01000010 00000010  0359        AND    R2,R2
0375  11000111 00100001  0360        XMIT   IVL,33                       READ HEADER SECTOR
0376  01010111 00000000  0361        AND    IV,AUX
0377  01100010 00000000  0362        XOR    R2,AUX                       COMPARE SECTOR #
0400  10100000 00000010  0363        NZT    AUX,READH5                   NOT EQUAL
0401  11100001 00000100  0364        JMP    READH6
0402  11000111 00000010  0365 READH5 XMIT   IVL,2                        SET IV BYTE 2
0403  11010011 00100001  0366        XMIT   IV(3,1),1                    ... BIT 3 TO 1
0404  ........ ........  0367 READH6 EQU    *
0404  11000000 00000111  0368        XMIT   AUX,7                        HEAD# MASK
0405  11000111 00010001  0369        XMIT   IVL,17                       SELECT SELECT CTL BYTE
0406  01010011 01100001  0370        AND    IV(3,3),R1                   MASK OUT CONTRL. HEAD#
0407  11000111 00100001  0371        XMIT   IVL,33                       SELECT DATA BYTE
0410  01010010 01100000  0372        AND    IV(2,3),AUX                  MASK OUT HEADER HEAD#
0411  01100001 00000001  0373        XOR    R1,R1                        COMPARE
0412  10100001 00001100  0374        NZT    R1,READH7                    NOT EQUAL
0413  11100001 00001110  0375        JMP    READH8
0414  11000111 00000010  0376 READH7 XMIT   IVL,2                        SET IV BYTE 2...
0415  11010001 00100001  0377        XMIT   IV(1,1),1                    ... BIT 1 FOR HEAD# ERROR
0416  11000111 00100100  0378 READH8 XMIT   IVL,36                       CHECK FOR END OF TRANSFER
0417  10010110 00101111  0379        XEC    IV(6,1),*
0420  11000001 00010100  0380        XMIT   R1,20                        WAIT 10 US FOR CRC
0421  11000000 11111111  0381        XMIT   AUX,255
0422  00100001 00000001  0382        ADD    R1,R1
0423  10100001 00010010  0383        NZT    R1,*-1
0424  11000111 00100101  0384        XMIT   IVL,37                       STOP TRANSFER
0425  11010001 00100000  0385        XMIT   IV(1,1),0
```

```
OBJ LOC   OBJECT WORD        SOURCE NBR  SOURCE LINE                          COMMENT
0426  11000111 00100100  0386          XMIT    IVL,36          MOVE HEADER CRC STATUS
0427  00010101 00100010  0387          MOVE    IV(5,1),R2      TO R2
0430  11000111 00000010  0388          XMIT    IVL,2           MOVE IT TO IV BYTE 2, BIT 4
0431  00000010 00110100  0389          MOVE    R2,IV(4,1)
0432  00000101 00001001  0390          MOVE    R5,R11          RESTORE RETURN VECTOR
0433  10001001 00011011  0391          XEC     R11,*           RETURN TO CALLER
0434  11100000 00010011  0392          JMP     COMM            RETURN 1 = SEEK
0435  11100000 01001000  0393          JMP     SECT3           RETURN 2 = SECTOR
0436  11100000 10100100  0394          JMP     FORM20          RETURN 3 = FORMAT
0437  ........ ........  0395  */PAGE/
0437  ........ ........  0396  *
0437  ........ ........  0397  **      READ SECTOR SUBROUTINE
0437  ........ ........  0398  *
0437  ........ ........  0399  READSUB EQU     *
0437  11000100 00011000  0400          XMIT    R4,24           LOW ORDER FOR 257+CRC
0440  11000011 00001000  0401          XMIT    R3,8            HIGH ORDER
0441  11001001 00000001  0402          XMIT    R11,1           RETURN VECTOR
0442  11100001 11001100  0403          JMP     DDSET
0443  11010010 00100001  0404  RDS1    XMIT    IV(2,1),1       READ MODE
0444  11000001 00010100  0405          XMIT    R1,20           COUNT 1/2 HEADER GAP (10 US)
0445  ........ ........  0406  RDS11   EQU     *
0445  11000000 11111111  0407          XMIT    AUX,255
0446  00100001 00000001  0408          ADD     R1,R1
0447  10100001 00100110  0409          NZT     R1,*-1
0450  11000111 00100101  0410          XMIT    IVL,37          START TRANSFER
0451  11010001 00100001  0411          XMIT    IV(1,1),1
0452  11000111 00100100  0412          XMIT    IVL,36          WAIT ON DATA READY
0453  10010111 00101011  0413          XEC     IV(7,1),*
0454  11000111 00100101  0414          XMIT    IVL,37          STROBE IT OUT (SYNC BYTE)
0455  11010110 00100001  0415          XMIT    IV(6,1),1
0456  11010110 00100000  0416          XMIT    IV(6,1),0
0457  11000000 00000001  0417          XMIT    AUX,1
0460  ........ ........  0418  RDS2    EQU     *
0460  11000111 00100100  0419          XMIT    IVL,36          READ TRANSFER STATUS
0461  10010111 01010001  0420          XEC     IV(7,2),*       WAIT LOOP & BRANCH TABLE
```

```
OBJ       OBJECT WORD       SOURCE          SOURCE LINE                    COMMENT
LOC                         NBR

0462  11100001 00110101  0421          JMP     RDS3              DATA READY
0463  11100001 00111101  0422          JMP     RDS4              END OF TRANSFER
0464  11100001 00110101  0423          JMP     RDS3              DATA READY & END OF TRANSFER
0465  ........ ........  0424  RDS3    EQU     *
0465  00000001 00001111  0425          MOVE    R1,IVR
0466  11000111 00100001  0426          XMIT    IVL,33            TRANSFER TO WS
0467  00010111 00011111  0427          MOVE    IV,WS
0470  00100001 00000001  0428          ADD     R1,R1             MOVE WS PTR
0471  11000111 00100101  0429          XMIT    IVL,37            STROBE IT OUT
0472  11010110 00100001  0430          XMIT    IV(6,1),1
0473  11010110 00100000  0431          XMIT    IV(6,1),0
0474  10100001 00110000  0432          NZT     R1,RDS2           NOT DONE = CONTINUE
0475  ........ ........  0433  RDS4    EQU     *
0475  11000001 00010100  0434          XMIT    R1,20             WAIT 10 US FOR CRC
0476  11000000 11111111  0435          XMIT    AUX,255
0477  00100001 00000001  0436          ADD     R1,R1
0500  10100001 00111111  0437          NZT     R1,*-1
0501  11000111 00100101  0438          XMIT    IVL,37            STOP TRANSFER
0502  11010001 00100000  0439          XMIT    IV(1,1),0
0503  11000111 00100100  0440          XMIT    IVL,36            MOVE SECTOR CRC STATUS
0504  00010101 00100010  0441          MOVE    IV(5,1),R2
0505  11000111 00000010  0442          XMIT    IVL,2             TO IV BYTE 2, BIT 5
0506  00000010 00110101  0443          MOVE    R2,IV(5,1)
0507  11100000 00010011  0444          JMP     COMM              RETURN
0510  ........ ........  0445  */PAGE/
0510  ........ ........  0446  *
0510  ........ ........  0447  **      WRITE HEADER SUBROUTINE
0510  ........ ........  0448  **      (USED ONLY WITH FORMAT COMMAND)
0510  ........ ........  0449  **      WRITES PREABLE (26 BYTES OF ZEROS),
0510  ........ ........  0450  **      SYNC BYTE (ALL ONES),
0510  ........ ........  0451  **      CYL ADDR (2 BYTES, MSB FIRST),
0510  ........ ........  0452  **      HEAD+SECTOR # (1 BYTE),
0510  ........ ........  0453  **      HEADER CRC (2 BYTES),
0510  ........ ........  0454  **      HEADER GAP (13 BYTES, ALL ZEROS)
0(  ) ........ ........  0455  **
```

```
OBJ LOC   OBJECT WORD          SOURCE NBR   SOURCE LINE                    COMMENT
0510  ........ ........  0456  **       R6 = SECTOR #
0510  ........ ........  0457  *
0510  ........ ........  0458  WRITHED  EQU     *
0510  11000100 00100000  0459           XMIT    R4,32          LOW ORDER FOR HEADER
0511  11000011 00000000  0460           XMIT    R3,0           HIGH ORDER
0512  11001001 00000101  0461           XMIT    R11,5          RETURN VECTOR
0513  11000000 00000000  0462           XMIT    AUX,0          WRITE MODE
0514  11100001 11001100  0463           JMP     DDSET
0515  11100001 11101010  0464  WRITH1   JMP     SECTW
0516  11000111 00010010  0465  WRITH2   XMIT    IVL,18         START WRITE
0517  11010011 00100000  0466           XMIT    IV(3,1),0
0520  11000000 11111111  0467           XMIT    AUX,255
0521  11000111 00100010  0468           XMIT    IVL,34
0522  00000000 00010111  0469           MOVE    AUX,IV         WRITE SYNC BYTE
0523  11000111 00100101  0470           XMIT    IVL,37         STROBE IT IN
0524  11010110 00100001  0471           XMIT    IV(6,1),1
0525  11010110 00100000  0472           XMIT    IV(6,1),0
0526  11000100 10100000  0473           XMIT    R4,160         DELAY 80 US FOR PREAMBLE
0527  00100100 00000100  0474           ADD     R4,R4
0530  10100100 01010111  0475           NZT     R4,*-1
0531  11000111 00100101  0476           XMIT    IVL,37         START TRANSFER
0532  11010001 00100001  0477           XMIT    IV(1,1),1
0533  11000000 00000001  0478           XMIT    AUX,1          CYL ADDR MSB MASK
0534  11000111 00010010  0479           XMIT    IVL,18
0535  01010111 00000001  0480           AND     IV,R1
0536  01100001 00000001  0481           XOR     R1,R1
0537  11000111 00100010  0482           XMIT    IVL,34         TRANSFER CYL ADDR MSB
0540  00000001 00010111  0483           MOVE    R1,IV
0541  11000111 00100101  0484           XMIT    IVL,37         STROBE IT IN
0542  11010110 00100001  0485           XMIT    IV(6,1),1
0543  11010110 00100000  0486           XMIT    IV(6,1),0
0544  11000000 11111111  0487           XMIT    AUX,255        INVERT CYL LSB
0545  11000111 00010011  0488           XMIT    IVL,19
0546  01110111 00000001  0489           XOR     IV,R1
0547  11000111 00100010  0490           XMIT    IVL,34         TRANSFER CYL ADDR LSB
```

```
OBJ LOC   OBJECT WORD        SOURCE NBR   SOURCE LINE                    COMMENT
0550  00000001 00010111  0491          MOVE    R1,IV
0551  11000111 00100101  0492          XMIT    IVL,37                 STROBE IT IN
0552  11010110 00100001  0493          XMIT    IV(6,1),1
0553  11010110 00100000  0494          XMIT    IV(6,1),0
0554  11000000 00000111  0495          XMIT    AUX,7                  MASK FOR HEAD#
0555  11000111 00010001  0496          XMIT    IVL,17                 SELECT SELECT CTL BYTE
0556  01010011 01100000  0497          AND     IV(3,3),AUX            MASK OUT HEAD#
0557  00000000 01100000  0498          MOVE    AUX(3),AUX             MOVE TO 3 MSB
0560  00100110 00000001  0499          ADD     R6,R1                  ADD IN WITH SECTOR#
0561  11000111 00100010  0500          XMIT    IVL,34                 TRANSFER SECTOR#/HEAD#
0562  00000001 00010111  0501          MOVE    R1,IV
0563  11000111 00100101  0502          XMIT    IVL,37                 STROBE IT IN
0564  11010110 00100001  0503          XMIT    IV(6,1),1
0565  11010110 00100000  0504          XMIT    IV(6,1),0
0566  11000111 00100100  0505          XMIT    IVL,36                 WAIT UNTIL CRC TRANSFERRED
0567  10010110 00110111  0506          XEC     IV(6,1),*
0570  11000000 11111111  0507          XMIT    AUX,255                DELAY FOR HEADER GAP
0571  11000100 01010000  0508          XMIT    R4,80
0572  00100100 00000100  0509          ADD     R4,R4
0573  10100100 01111010  0510          NZT     R4,*-1
0574  11000111 00010010  0511          XMIT    IVL,18                 STOP WRITING
0575  11010011 00100001  0512          XMIT    IV(3,1),1
0576  11000111 00100101  0513          XMIT    IVL,37
0577  11010001 00100000  0514          XMIT    IV(1,1),0              STOP TRANSFER
0600  11001001 00000010  0515          XMIT    R11,2                  TRANSFER VECTOR WITHIN FORMAT
0601  11100000 10101000  0516          JMP     FORM21                 RETURN TO FORMAT
0602  ........ ........  0517  */PAGE/
0602  ........ ........  0518  *
0602  ........ ........  0519  **      SUBROUTINE TO WRITE SECTOR
0602  ........ ........  0520  **      APPENDS SYNC BYTE +256 DATA BYTES +CRC BYTES TO HEADER
0602  ........ ........  0521  *
0602  ........ ........  0522  WRITSUB EQU     *
0602  11000111 00000010  0523          XMIT    IVL,2                  CHECK FOR HEADER ERROR
0603  11000000 00011011  0524          XMIT    AUX,27                 HD#/SECT/CRC/WPROT
060   01010100 10000000  0525          AND     IV(4,4),AUX
```

```
OBJ LOC   OBJECT WORD   SOURCE NBR  SOURCE LINE                          COMMENT
0605  10100000 10001011  0526          NZT     AUX,WRERR       IF ERROR, DON'T WRITE
0606  11000100 00001000  0527          XMIT    R4,8            GET...
0607  11000011 00001000  0528          XMIT    R3,8            ...LENGTH
0610  11001001 00000010  0529          XMIT    R11,2           GET RETURN VECTOR
0611  11000000 00000000  0530          XMIT    AUX,0           WRITE MODE
0612  11100001 11001100  0531          JMP     DDSET
0613  11100000 00010011  0532  WRERR   JMP     COMM
0614  ........ ........  0533  WRS1    EQU     *
0614  11000111 00010010  0534          XMIT    IVL,18          START WRITE
0615  11010011 00100000  0535          XMIT    IV(3,1),0
0616  11000001 11111111  0536          XMIT    R1,255          WRITE SYNC BYTE
0617  11000111 00100010  0537          XMIT    IVL,34
0620  00000001 00010111  0538          MOVE    R1,IV
0621  11000111 00100101  0539          XMIT    IVL,37          STROBE IT IN
0622  11010110 00100001  0540          XMIT    IV(6,1),1
0623  11010110 00100000  0541          XMIT    IV(6,1),0
0624  11000000 11111111  0542          XMIT    AUX,255         WAIT HEADER GAP TIME
0625  11000100 00100001  0543          XMIT    R4,33
0626  00100100 00000100  0544          ADD     R4,R4
0627  10100100 10010110  0545          NZT     R4,*-1
0630  11000000 00000001  0546          XMIT    AUX,1
0631  11000111 00100101  0547          XMIT    IVL,37
0632  11010001 00100001  0548          XMIT    IV(1,1),1       START TRANSFER
0633  ........ ........  0549  WRS2    EQU     *
0633  00100001 00000001  0550          ADD     R1,R1           MOVE WS PTR
0634  00000001 00001111  0551          MOVE    R1,IVR
0635  11000111 00100010  0552          XMIT    IVL,34
0636  00011111 00010111  0553          MOVE    WS,IV           TRANSFER BYTE
0637  11000111 00100101  0554          XMIT    IVL,37
0640  11010110 00100001  0555          XMIT    IV(6,1),1       STROBE IT IN
0641  11010110 00100000  0556          XMIT    IV(6,1),0
0642  11000111 00100100  0557          XMIT    IVL,36          TEST NEXT
0643  ........ ........  0558  *
0643  10010111 01000011  0559  WRS3    XEC     IV(7,2),*       WAIT LOOP & BRANCH TABLE
0644  11100001 10011011  0560          JMP     WRS2            DATA READY
```

```
OBJ       OBJECT WORD      SOURCE        SOURCE LINE                COMMENT
LOC                        NBR

0645  11100001 10100111   0561          JMP     WRS4               TRANSFER END
0646  11100001 10100111   0562          JMP     WRS4               TRANSFER END & MORE DATA
0647  ........ ........   0563  WRS4    EQU     *
0647  11000000 11111111   0564          XMIT    AUX,255            DELAY FOR POSTAMBLE
0650  11000100 00101000   0565          XMIT    R4,40
0651  00100100 00000100   0566          ADD     R4,R4
0652  10100100 10101001   0567          NZT     R4,*-1
0653  11000111 00010010   0568          XMIT    IVL,18             STOP WRITING
0654  11010011 00100001   0569          XMIT    IV(3,1),1
0655  11000111 00100101   0570          XMIT    IVL,37             STOP TRANSFER
0656  11010001 00100000   0571          XMIT    IV(1,1),0
0657  11100000 00010011   0572          JMP     COMM               RETURN TO COMMAND
0660  ........ ........   0573  */PAGE/
0660  ........ ........   0574  *
0660  ........ ........   0575  **      SELECT DRIVE SUBROUTINE
0660  ........ ........   0576  *
0660  00000001 01000001   0577  SELECT  MOVE    R1(2),R1           GET DRIVE # IN 2 LOW BITS
0661  11000000 00000011   0578          XMIT    AUX,3              MASK OFF REST OF R1
0662  01000001 00000000   0579          AND     R1,AUX
0663  11000111 00010001   0580          XMIT    IVL,17             SELECT SELECT CTL WORD
0664  10000000 11000101   0581          XEC     AUX,SELTAB         SET THE BITS
0665  11000000 00000100   0582          XMIT    AUX,4              INVERSION MASK
0666  01100010 10100000   0583          XOR     R2(5),AUX          GET HEAD#
0667  00000000 01110011   0584          MOVE    AUX,IV(3,3)        SEND TO INTERFACE
0670  11000111 00010101   0585          XMIT    IVL,21             GET WRITE PROTECT BIT...
0671  00010010 00100000   0586          MOVE    IV(2,1),AUX
0672  11000111 00000010   0587          XMIT    IVL,2              ...AND MOVE TO ERR BYTE
0673  00000000 00110000   0588          MOVE    AUX,IV(0,1)        ...HIGH BIT
0674  ........ ........   0589  *
0674  ........ ........   0590  *
0674  11000111 00010101   0591          XMIT    IVL,21             CHECK FOR DRIVE READY
0675  10110000 00111111   0592          NZT     IV(0,1),SELRET
0676  11100001 11001001   0593          JMP     ERROR
0677  10001001 10111111   0594  SELRET  XEC     R11,*              RETURN
0700  11100000 00110100   0595          JMP     SEEK1              RETURN 1 = SEEK
```

```
OBJ LOC   OBJECT WORD      SOURCE NBR  SOURCE LINE                          COMMENT

0701  11100000 01000110   0596        JMP     SECT2                        2 = SECTOR
0702  11100000 10010011   0597        JMP     FORMS                        3 = FORMAT
0703  11100000 10001010   0598        JMP     READU1                       4 = READ UNFORMATTED
0704  11100000 01110110   0599        JMP     STAT1                        5 = STATUS
0705  ........ ........   0600  SELTAB  EQU     *
0705  11010111 10001110   0601        XMIT    IV(7,4),14                   DRIVE 1
0706  11010111 10001101   0602        XMIT    IV(7,4),13                   DRIVE 2
0707  11010111 10001011   0603        XMIT    IV(7,4),11                   DRIVE 3
0710  11010111 10000111   0604        XMIT    IV(7,4),7                    DRIVE 4
0711  ........ ........   0605  *
0711  11000111 00000010   0606  ERROR   XMIT    IVL,2                        BIT 7 OF IV-BYTE 2 ...
0712  11010111 00100001   0607        XMIT    IV(7,1),1                    ... SET FOR "DRIVE NOT READY"
0713  11100000 00010011   0608        JMP     COMM
0714  ........ ........   0609  */PAGE/
0714  ........ ........   0610  *
0714  ........ ........   0611  **      DDSET -- SET UP DISK DATA BOARD
0714  ........ ........   0612  *
0714  ........ ........   0613  DDSET   EQU     *
0714  11000111 00100101   0614        XMIT    IVL,37                       SET UP DISK DATA CONTROLS
0715  00000000 00110010   0615        MOVE    AUX,IV(2,1)                  SET READ/WRITE MODE
0716  11010001 01000010   0616        XMIT    IV(1,2),2                    SET CRCAPE, NO START
0717  11010111 10000000   0617        XMIT    IV(7,4),0                    SET SMS, NO ACK, CLEAR
0720  11010100 00100001   0618        XMIT    IV(4,1),1                    RESET CLEAR
0721  11000111 00100011   0619        XMIT    IVL,35                       SELECT CONTROL BYTE
0722  11010011 10001111   0620        XMIT    IV(3,4),15                   RESET LOAD STROBES
0723  00000100 10010111   0621        MOVE    R4,IV(7,4)
0724  11010011 00100000   0622        XMIT    IV(3,1),0                    NIBBLE 1
0725  11010011 00100001   0623        XMIT    IV(3,1),1
0726  00000100 10000100   0624        MOVE    R4(4),R4                     ROTATE TO GET NEXT
0727  00000100 10010111   0625        MOVE    R4,IV(7,4)
0730  11010010 00100000   0626        XMIT    IV(2,1),0                    NIBBLE 2
0731  11010010 00100001   0627        XMIT    IV(2,1),1
0732  00000011 10010111   0628        MOVE    R3,IV(7,4)
0733  11010001 00100000   0629        XMIT    IV(1,1),0                    NIBBLE 3
0734  11010001 00100001   0630        XMIT    IV(1,1),1
```

```
OBJ LOC   OBJECT WORD    SOURCE NBR   SOURCE LINE                    COMMENT
0735  00000011 10000011  0631         MOVE   R3(4),R3               ROTATE TO GET NEXT
0736  00000011 10010111  0632         MOVE   R3,IV(7,4)
0737  11010000 00100000  0633         XMIT   IV(0,1),0               NIBBLE 4
0740  11010000 00100001  0634         XMIT   IV(0,1),1
0741  11000111 00100101  0635         XMIT   IVL,37                  LEAVE SETUP FOR R/W MODE SELECT
0742  11010100 00100000  0636         XMIT   IV(4,1),0               SET CLEAR
0743  11010100 00100001  0637         XMIT   IV(4,1),1               RESET CLEAR
0744  10001001 11100100  0638         XEC    R11,*                   RETURN TO CALLER
0745  11100001 00100011  0639         JMP    RDS1                    RETURN 1 = READSUB
0746  11100001 10001100  0640         JMP    WRS1                           2 = WRITSUB
0747  11100000 11010000  0641         JMP    READH1                         3 = READHED
0750  11100000 10001110  0642         JMP    READU2                         4 = READ UNFORMATTED
0751  11100001 01001101  0643         JMP    WRITH1                         5 = WRITHED
0752  ........ ........  0644  */PAGE/
0752  ........ ........  0645  *
0752  ........ ........  0646  **     SECTOR WAIT SUBROUTINE
0752  ........ ........  0647  *
0752  ........ ........  0648  SECTW  EQU    *
0752  11000011 00011000  0649         XMIT   R3,24                   COUNT DOWN 24 SECTOR PULSES
0753  11000111 00010110  0650         XMIT   IVL,22                  WAIT ON SECTOR PULSE...
0754  10110000 00101100  0651  SECTWO NZT    IV(0,1),*               ...TO GO LOW
0755  10010000 00101101  0652         XEC    IV(0,1),*               ...AND THEN HIGH
0756  11000000 00011111  0653         XMIT   AUX,31                  GET 5 BIT MASK
0757  01000010 00000000  0654         AND    R2,AUX                  MASK OFF SECTOR
0760  01110111 10100000  0655         XOR    IV(7,5),AUX             COMPARE TO SECTOR COUNTER
0761  10100000 11110110  0656         NZT    AUX,SECTW1              IF DIFF., TRY AGAIN
0762  10001001 11110000  0657         XEC    R11,*-2                 RETURN TO CALLER
0763  11100000 11010001  0658         JMP    READH2                  3 = READHED
0764  11100000 10001111  0659         JMP    READU3                  4 = READUN
0765  11100001 01001110  0660         JMP    WRITH2                  5 = WRITHED
0766  ........ ........  0661  *
0766  11000000 11111111  0662  SECTW1 XMIT   AUX,255                 DECREMENT SECTOR COUNT
0767  00100011 00000011  0663         ADD    R3,R3
0770  10100011 11101100  0664         NZT    R3,SECTWO               MORE TO GO ?
0771  11000111 00000010  0665         XMIT   IVL,2                   SET BIT 6 OF IV-BYTE ?
```

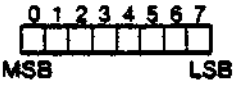
```
OBJ LOC   OBJECT WORD        SOURCE NBR   SOURCE LINE              COMMENT

0772  11010110 00100001  0666          XMIT    IV(6,1),1        ... FOR "BAD SECTOR" ERROR
0773  11100000 00010011  0667          JMP     COMM
```

5. UNDERSTANDING THE 8X300 INSTRUCTION SET

DESCRIPTION

The 8X300 has a repertoire of 8 instruction classes which allow the user to test input status lines, set or reset output control lines, and perform high speed input/output data transfers. All instructions are 16 bits in length. Each instruction is fetched, decoded and executed completely in 250ns.

Data is represented as an 8-bit byte; bit positions are numbered from left to right, with the least significant bit in position 7.



Within the Interpreter, all operations are performed on 8-bit bytes. The interpreter performs 8-bit, unsigned 2's complement complement arithmetic.

INSTRUCTION FORMATS

The general 8X300 instruction format is:

0 1 2	3 4 5 6 7 8 9 10 11 12 13 14 15
Op Code	Operand(s)

Table 1 contains a summary of the 8X300 instruction set and description of the operand fields.

Instructions are specified by a 3-bit Operation (Op) Code field. The operand may consist of the following fields: Source (S) field, Destination (D) field, Rotate/Length (R/L) field, Immediate (I) Operand field, and (Program Storage) Address (A) field.

The instructions are divided into 5 format types based on the Op Code and the form of the Operand(s) as shown in Figure 1.

OPERATION	FORMAT	RESULT	NOTES
MOVE		Content of data field addressed by S, R/L replaces data in field specified by D, R/L.	If S and D both are register addresses then R L specifies a right rotate of R/L places applied to the register specified by S.
ADD		Sum of AUX and data specified by S, R/L replaces data in field specified by D, R/L.	
AND		Logical AND of AUX and data specified by S, R/L replaces data in field specified by D, R/L.	
XOR		Logical exclusive OR of AUX and data specified by S, R/L replaces data in field specified by D, R/L.	
XMIT		The literal value I replaces the data in the field specified by S, L.	If S is IV or WS address then I limited to range 00-37. Otherwise I limited to range 000-377.
NZT		If the data in the field specified by S, L equals zero, perform the next instruction in sequence. If the data specified by S,L is not equal to zero, execute the instruction at address determined by using the literal I as an offset to the Program Counter.	If S specifies an IV or WS address then I is limited to the range 00-37. I is limited to the range 000-377 otherwise.
XEC		Perform the instruction at address determined by applying the sum of the literal I and the data specified by S,L as an offset to the Program Counter. If that instruction does not transfer control, the program sequence will continue from the XEC instruction location.	The offset operation is performed by reducing the value of PC to the nearest multiple of 32 (if I : 00-37) or 256 (if I : 000-377) and adding the offset.
JMP		The literal value I replaces contents of the Program Counter.	I limited to the range 00000-07777.

Table 1 8X300 INSTRUCTION SUMMARY

INSTRUCTION FORMATS

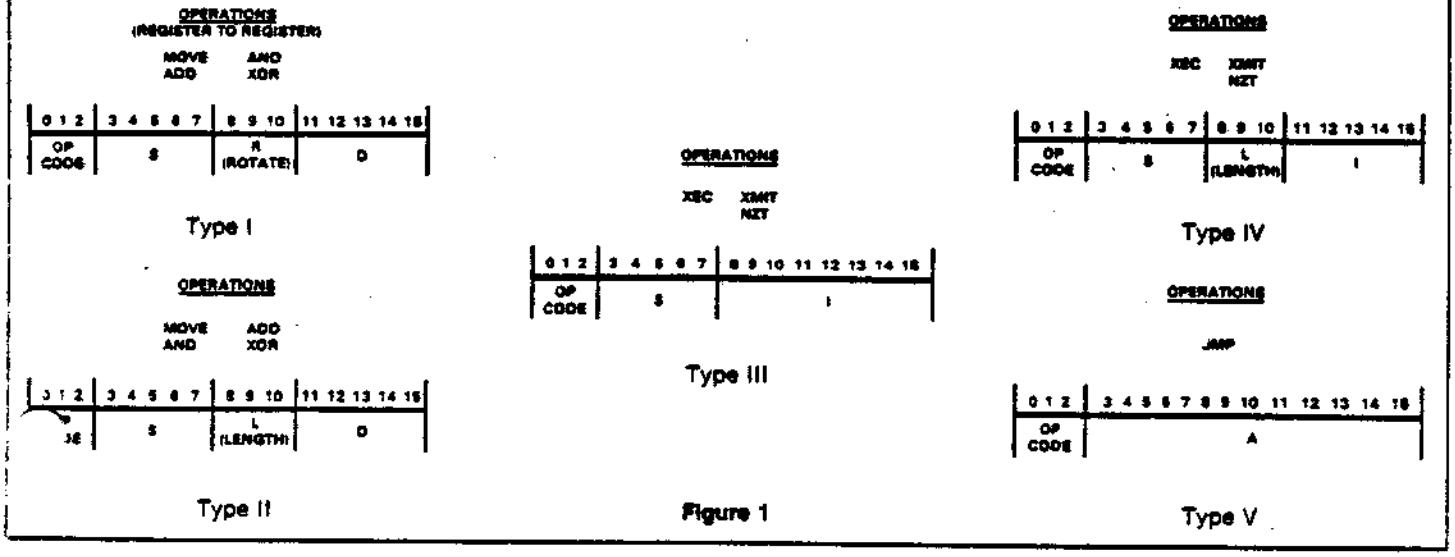
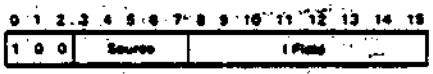


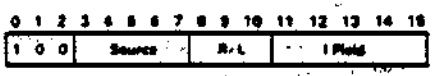
Figure 1

XEC I(S)

Format



Type III



Type IV

Operation

Execute instruction at the address specified by the Address Register with lower 5/8 bits replaced by (S) + I.

Description

Execute the instruction at the address determined by replacing the low order bits of the Address Register (AR) with the low order bits of the sum of the literal I and the contents of the source field. If S is a register, the low order 8 bits of AR are replaced; if S is an IV or Working Storage field, the low order 5 bits of AR are replaced, resulting in an execute range of 256 and 32 respectively. The Program Counter is not affected unless the instruction executed is a JMP or NZT (whose branch is taken).

Example

Execute one of n JMPs in a table of JMP instructions determined by the value of the selected IV byte field. The table follows immediately after the XEC instruction and the IV field is called INTERPT and is a 3-bit field located in bits 4, 5 and 6.

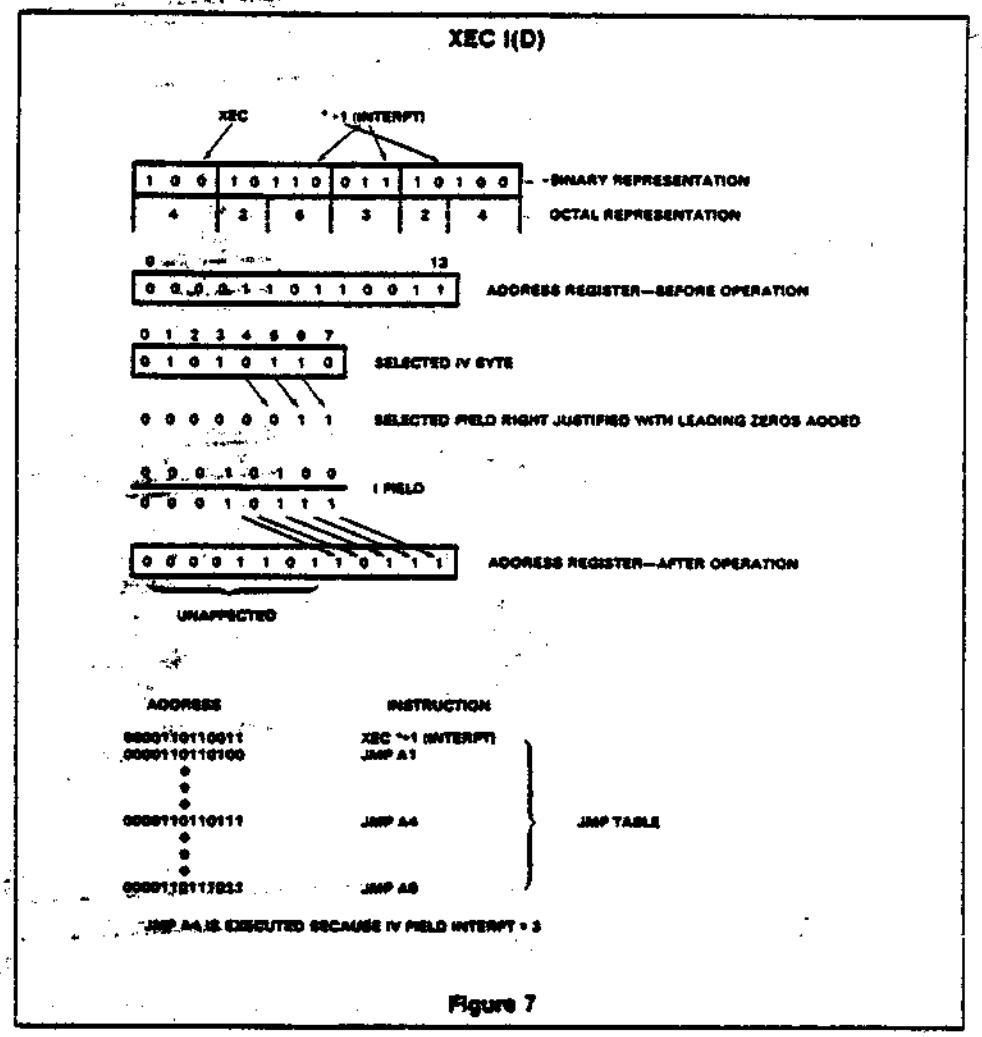
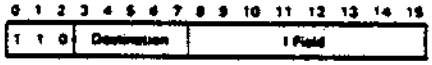


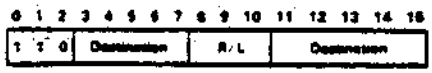
Figure 7

XMIT I,D

Format



Type III



Type IV

Operation: I→(D)

Description

Transmit literal. The literal field I is stored in D. If D is a register, an 8-bit field is transferred; if D is an IV or Working Storage field, up to a 5-bit field is transferred.

Example

Store the bit pattern 110 in the selected Working Storage field. The field name is VALUE and is located in bits 3, 4 and 5.

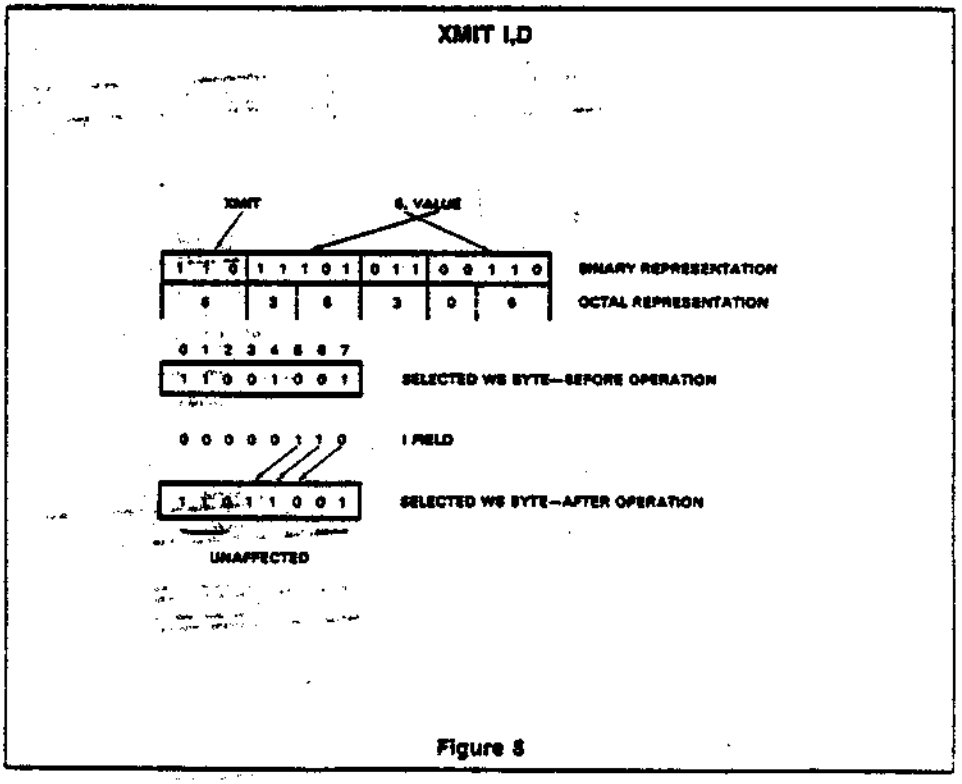
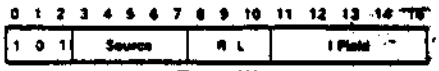


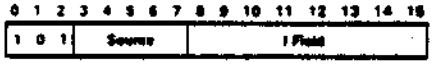
Figure 8

ZT S,I

Format



Type III



Type IV

Operation

Non-Zero Transfer. If (S) ≠ 0, PC offset by I → PC; otherwise PC + 1 → PC.

Description

If the data specified by the S field is non-zero, replace the low order bits of the Program Counter with I. Otherwise, processing continues with the next instruction in sequence. If S is a register, the low order 8 bits of the PC are replaced; if S is an IV or Working Storage field, the low order 5 bits of the PC are replaced, resulting in an NZT range of 256 and 32 respectively.

Example

...p to Program Address ALPHA if the ...ected IV byte field is non-zero. The field name is OVERFLO and it is a 1-bit field located in bit 3.

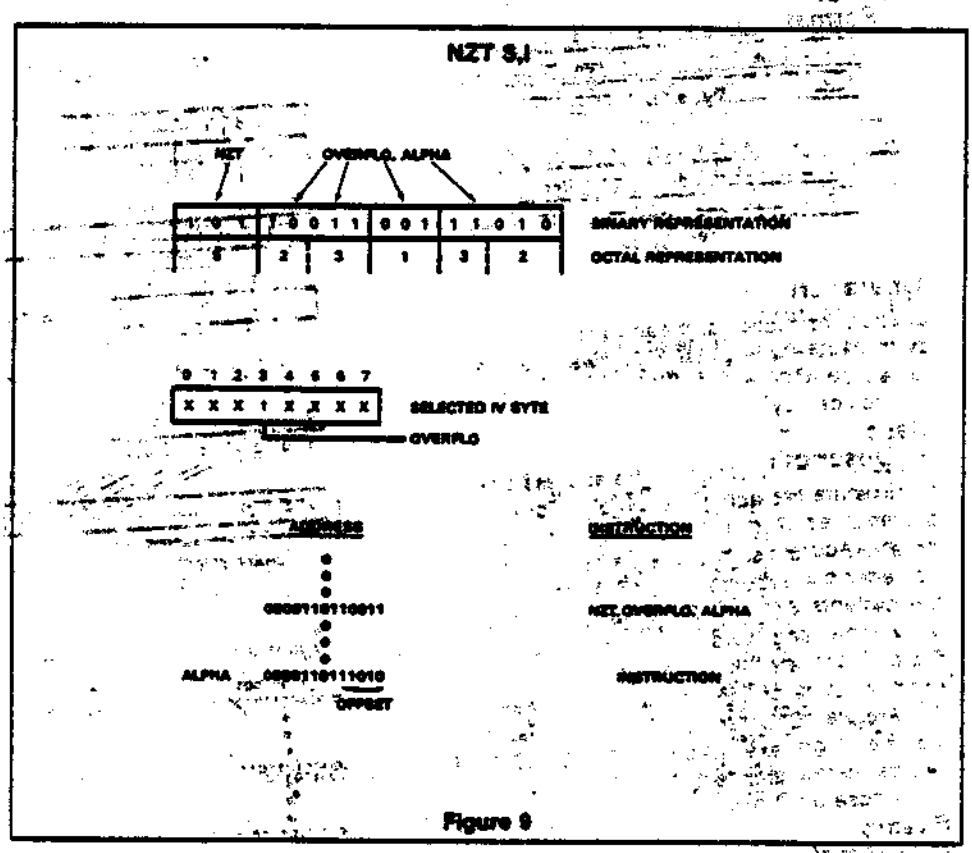
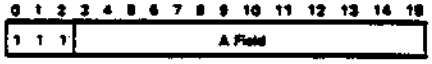


Figure 9

JMP A

Format: Type V



Operation: A → PC

Description

The literal value A is placed in the Program Counter and processing continues at location A. A has a range of 0-17777$_8$ in current systems (0-8191).

Example

Jump to location ALPHA (0000101110001)

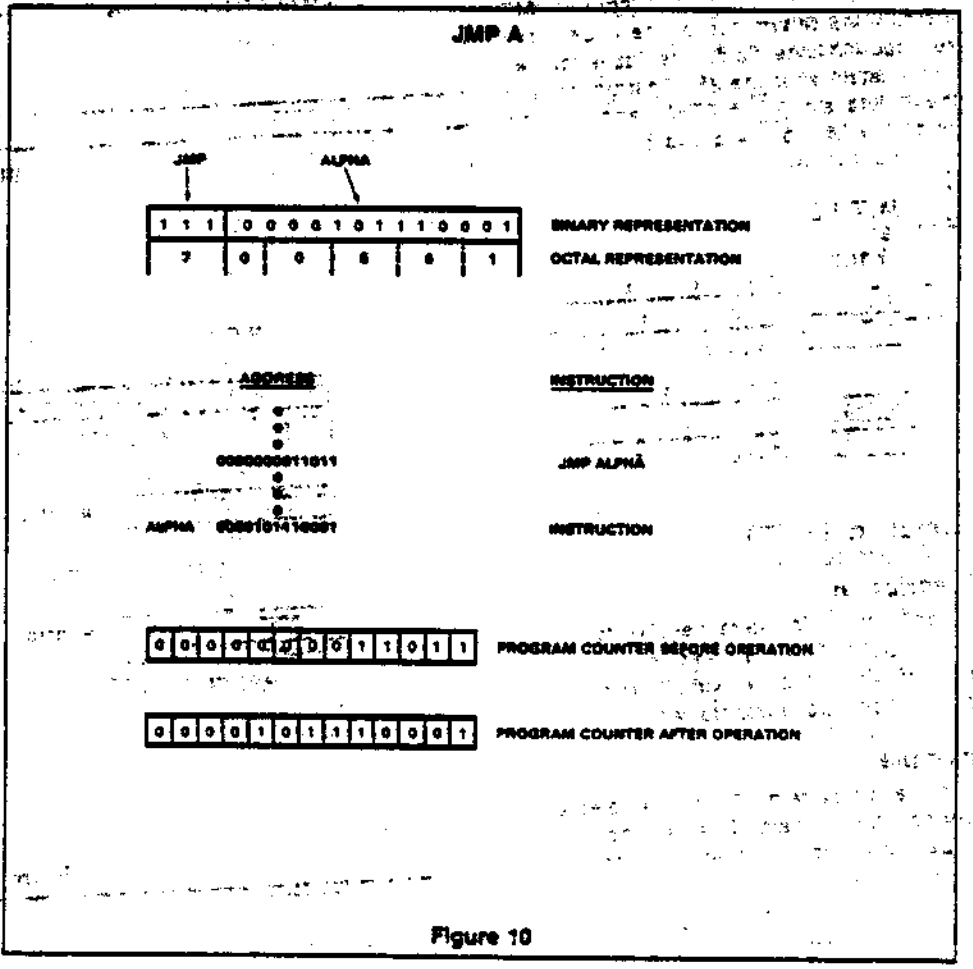


Figure 10